MTR100302
MITRE TECHNICAL REPORT

MITRE

Reliable Electronic Text: The Elusive Prerequisite for a Host of Human Language Technologies

Paul M. Herceg
Catherine N. Ball
September 30, 2010

Contract No.: W15P7T-10-C-F600
Project No.: 0710G01Z-HB

Report Documentation Page		*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE **30 SEP 2010**	2. REPORT TYPE	3. DATES COVERED **00-00-2010 to 00-00-2010**
4. TITLE AND SUBTITLE **Reliable Electronic Text: The Elusive Prerequisite for a Host of Human Language Technologies**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **MITRE Corporation,7525 Colshire Drive,McLean,VA,22102**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Electronic text for use by human language technologies originates from a number of sources direct keyboard entry, optical character recognition, speech recognition, and text-containing computer files. In particular, text-containing computer files may elude processing by an array of human language technology applications (e.g., search, language ID, machine translation, and text analytics). This paper brings to light the effort required to extract electronic text from these files preserve its integrity, and, for some use cases, preserve its structure. It explores a series of specific human language technologies, highlighting the following aspects for each: relevant use cases, the impact of text extraction or conversion errors, the criticality of dependable text extraction and reliable electronic text, and the importance of experimentation and/or testing prior to use. Overall, this paper promotes the successful use of human language technology by equipping the reader to be discerning about the use of human language technology applications with text-containing files.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**13**	

Abstract

Electronic text for use by human language technologies originates from a number of sources: direct keyboard entry, optical character recognition, speech recognition, and text-containing computer files. In particular, text-containing computer files may elude processing by an array of human language technology applications (e.g., search, language ID, machine translation, and text analytics). This paper brings to light the effort required to extract electronic text from these files, preserve its integrity, and, for some use cases, preserve its structure. It explores a series of specific human language technologies, highlighting the following aspects for each: relevant use cases, the impact of text extraction or conversion errors, the criticality of dependable text extraction and reliable electronic text, and the importance of experimentation and/or testing prior to use. Overall, this paper promotes the successful use of human language technology by equipping the reader to be discerning about the use of human language technology applications with text-containing files.

Keywords: Electronic text, human language technology, computer files, search, language ID, machine translation, text analytics, content extraction.

Table of Contents

1. Introduction

Electronic text is a prerequisite for text-processing applications such as indexing and search, named entity recognition (NER), and machine translation (MT). For example, text that is printed on paper must first be converted to electronic form to make it searchable—typically, by scanning the pages and using Optical Character Recognition (OCR) to create electronic text.

Not all electronic text is created equal, however. Electronic text comes in a wide variety of containers, from Microsoft Office documents to Oracle database records. Electronic text comes in a wide variety of character set encodings, such as Chinese “Big 5” or Unicode UTF-8. Electronic text may be written in any of the world’s languages, including the special sublanguages of blogs, chat and forums. And from the end user’s perspective, even formatting aspects of the electronic text may be important: for example, if the use case is translating a document for a customer, and the customer requires the translated document to be formatted to look like the original.

From the application perspective, all aspects of electronic text potentially make a difference. Almost all products are language-specific, and will produce useful results only on supported languages. In terms of file types, some products accept Microsoft Word documents as input, while others may accept only plain text files. Some products may work best if all text is “normalized” to a single character set encoding such as UTF-8.

The purpose of this paper is to explore factors to be considered when trying to match up text-containing files (in a variety of file formats, character set encodings, languages, etc.) with text-processing applications and use cases.

2. Electronic Text and Search Engines

We explore the factors for matching up electronic text with applications by discussing major application areas. Because most readers are familiar with Web search engines, we first discuss enterprise search. There are several commercial and open source options for enterprise search, including Endeca Information Access Platform, Microsoft FAST Search Server 2010 for SharePoint, Solr/Lucene, Oracle Secure Enterprise Search, and Google Search Appliance. The following are typical use cases for search:

- Use case: Allow a user to find documents matching a query—in near-real-time
- Use case: Allow a user to browse the documents of a collection through iterative queries—in near-real-time

The state-of-the-practice approach for meeting the near-real-time aspect of the search use case is to create an inverted index and allow querying against that index. Search engine indexers need access to the electronic text inside each file of the document collection in order to create the inverted index. However, when the electronic text to be searched resides inside files in any number of *rich media* file types (i.e., files other than plain text; Gospodnetić & Hatcher, 2005, p. 224), a significant amount of preprocessing is needed to free, or extract, the electronic text from each file format.

From one perspective, rich media file formats must be converted to a normalized form. A file format parser is the tool to perform this preprocessing. Search engines are typically bundled with a number of file format parsers, or with instructions on how to obtain or create file format parsers. This is due to a couple of factors: the search engine relies on electronic text, and the use case typically involves indexing the electronic text originating from a variety of file types. Some vendors have directly addressed the need for extracting electronic text from a wide variety of file types and developed their own *content extraction software,* also known as *content extractors* or *text extractors*. These applications provide a single interface to vast collections of file format parsers. We turn to a more detailed treatment of content extraction in the next section.

3. Electronic Text and Content Extraction

Content extraction software provides access to the textual content of a computer file, as a whole, without discarding textual content (Herceg, 2009). It involves identifying the existing file format of the file (called *file typing*), applying an algorithm for decoding the content, and normalizing the content to a desired output file format or data object (Herceg, Allison, & Ball, 2009). For the extracted text to be *reliable electronic text*, it must also preserve the reading order and integrity of textual objects from the original file. For a detailed explanation of content extraction technology, we refer the reader to Herceg (2009). Here is a typical content extraction use case:

- Use case: Extract electronic text from all files into a single, common electronic text form (i.e., a normalized form, in files or data objects), suitable for viewing with one or more applications, or suitable for processing by another application, such as a search engine.

Content extractors can be used on their own, for example to create linguistic corpora, or for preprocessing prior to any number of human language technology (HLT) applications, including enterprise search engines, pattern matching, named entity recognition, name matching, information extraction, text categorization, and text mining. Search engine vendors and other application vendors can find it cost effective to embed a best-of-breed, third party commercial content extractor and/or a set of open source file format parsers, rather than develop their own content extractor. Although the leading search engines embed content extractors and can extract and index electronic text from a wide variety of file types, the number of file formats handled is limited by the file format parsers hosted within the embedded content extraction software. For example, "many [file format parsers called] iFilters have shipped with the new versions of SharePoint [2010], including ascx, asm, asp, aspx, bat, c, cmd, cpp, css, cxx, dev, dic, doc, docm, docx, dot, eml, h, hhc, hht, hta, htm, html, htw, lnk, mht, mhtml, mpx, msg, odc, pot, pps, ppt, pptm, pptx, pub, stm, tif, trf, vsd, xlb, xlc, xls, xlsm, xlsx, xlt, and xml" (English, Alderman, & Ferraz, in press, p. 531).

Every human language technology that embeds a content extractor has a limit on the file types supported (i.e., the file format parsers provided). Therefore, it is important for a user of HLT software to investigate, understand, and consider these limits when applying an enterprise search product, or any other HLT software that embeds content extraction or file format parsers (e.g., named entity recognition, information extraction, machine translation, computer assisted translation, text categorization, and text mining).

A natural work around for the limitations of a given content extraction application is to use file conversion software. For example, if the content extractor does not support file format A, but does support a related file format B, the user can use a file converter to transform file *a* in file format A into file *b* in file format B. In most cases, the content extractor can process converted file *b*. However, there are limitations, and we consider these in the next section.

It is important to know that extracting electronic text is particularly challenging for some file formats, and we briefly highlight two here: HyperText Markup Language (HTML) and Portable Document Format (PDF). HTML is a syntax-constrained plain text file format. The syntax of an HTML file provides no reliable sequence of bytes for differentiating the HTML file type from other file types (Magic Number, 2009; File Format, 2010). This complicates the identification of the HTML file type, which is essential for text extraction.[1] A PDF file provides instructions for painting glyphs and other document objects on one or more pages (Adobe, 2008, p. vii; King, 2008).[2] Text painting instructions use strings of *character codes*, denoting glyphs in a font encoding, from which electronic text can be derived under certain conditions (Adobe, pp. 251, 292). A glyph's *character code* may be the code point value in a standard encoding such as ASCII or Unicode, but this is not always the case—for example, ligatures in Latin-fonts, and glyphs in Arabic fonts (King, 2008; Carrier, 2009). This complexity, and others, makes text extraction from PDFs challenging.[3]

4. Electronic Text and the Side Effects of Content Extraction and File Conversion

Freeing electronic text from rich media file formats, or transforming the rich media file format container, often comes at a cost. The file format parsers in a content extractor or file converter are not perfect and have side effects. Digital archivists have long been interested in the effect of file conversion (and normalization) on electronic text, referring to it as *file migration*. Waters & Garrett (1996) and Lawrence et al. (2000) discuss that file migration can compromise the content integrity of textual objects.[4] The following are example use cases that are of concern to digital archivists:

- Use case: Convert a file (or files) to a file format that is a more robust, stable, modern, or enduring (Lawrence et al., pp. 20-21; Waters & Garrett, p. 6)
- Use case: Convert a file (or files) to a smaller, more manageable set of file formats (Waters & Garrett, p. 28).

The side effects of content extraction and file conversion, both of which impact electronic text, are well documented in digital library literature. These effects include the following:

- *Mishandling or loss of structural elements* (Lawrence et al., p. 2, Waters & Garrett, p. 28).[5] For example, table text can be extracted without keeping together the text within each table cell. A

[1] Herceg (2009) explains that relying solely on the file extension is insufficient for file format identification (i.e., file typing).
[2] A glyph is the graphical shape of an abstract character in a given alphabet. On a computer system, glyphs are encoded in fonts. The shape of a given alphabetic character may change depending on its context.
[3] Other complexities include decoding strings of *character codes* using a character mapping (which may or may not be present), reforming words (e.g., words that were broken into two or more chunks, or that were hyphenated and broken across two lines), reconstituting newlines and reading order based on spatial or structural information, normalizing decomposable Unicode characters, and, converting right-to-left language from presentation order into storage order, while not affecting left-to-right components (King, 2008; Carrier, 2009). Depending on the use case, the latter may require language ID.
[4] Waters & Garrett (1996) explain that a change in media type can also compromise content integrity (e.g., digital to paper, paper to digital) (p.27).
[5] The formatting codes/tags in a Microsoft Word file or an HTML file are examples of structural elements.

column of numbers can be concatenated as a horizontal string of numbers. Headers and footers can interleave with a document's body text.
- *Mishandling or loss of character data* (Rauch & Rauber, 2004, p. 203). For example, some applications periodically drop characters because no character mapping exists, or fully garble the electronic text because of software bugs, such as failing to recognize and handle encoding, incorrectly applying hardware byte order read preference, etc.
- *Mishandling or loss of textual objects* (Lawrence et al., p. 2). For example, floating point numbers can become truncated. Spreadsheet cell values can be preserved, while cell formulas are dropped.

These side effects matter to the users and developers of HLT applications because such effects can render electronic text useless for natural language processing (NLP). Typically, file converters do not expose the details of the conversion algorithm (Lawrence et al., p. 8). Therefore it is important for the user of the content extraction or file conversion software to test and validate the performance of its text handling, and make note of any electronic text side effects. A converter's or extractor's electronic text side effects must be considered when determining the effectiveness of cascading the output to HLT applications such as search engines, pattern matching, named entity recognition, name matching, information extraction, text categorization, and text mining.

5. Electronic Text and Language ID

Identifying the language of electronic text, or language ID, is an important function when a source document collection is multilingual. The following are typical use cases for language ID software:
- Use case 1: The user has a plain text UTF-8 file, identify the language.
- Use case 2: The user has a set of plain text files, identify the language and encoding of each.
- Use case 3: The user has a set of files. Sort them by the dominant language of each file.
- Use case 4: The user has a set of files for search engine indexing. Sort them by dominant language, and index them. (In this case, language ID is fully integrated with a search engine.)
- Use case 5: The user has a set of files. Identify the starting and ending offsets for each language in each file for follow-on processing.

Ideally, language ID works on normalized electronic text (use case 1). The language ID software with which we are familiar also recognizes the character encoding (use case 2; See Condon, Herceg, Phillips & Rubenstein, 2005).[6] The identification is performed with heuristics in some cases, but state-of-the-art software applies pre-trained n-gram models to detect language. There may be a character quota for the language ID software to operate successfully on a given textual object. Beyond these simple use cases, language ID finds itself in the middle of more complicated use cases that require content extraction preprocessing (use cases 3, 4, and 5).[7]

Use cases 1, 2, 3, and 4 involve document-level language ID, whereas use case 5 involves more fine-grained language ID.

[6] If the language ID software does not detect encoding, then separate encoding detection software is required.
[7] Granted, language ID software may operate on a number of file types. For example, it may operate on rich media files for which the electronic text is not obfuscated, without performing any file format parsing. However, this does not obviate the need for content extraction preprocessing in order to free the textual objects from rich media file formats.

For search engine indexing of electronic text files in multiple languages, language ID is vital (use case 4). It allows the system administrator or software to apply the appropriate language-specific linguistic preprocessor in order to identify word boundaries (isolate words), handle word morphology, stem words, select stop words, and select words for indexing (Baeza-Yates & Ribeiro-Neto, 1999, p. 165).[8]

The reader should approach with caution any software solution that combines content extraction and language ID because of the risk of lost or garbled electronic text. This includes search engines. The user should confirm with the vendor that the solution includes encoding ID, which is essential for proper extraction of electronic text in the world's languages. Briefly turning back to the topic of content extraction, content extractors tend to naïvely apply file format parsing, garbling electronic text in any one of a plethora of known, standard single-byte or multi-byte text encodings. This is a complex issue discussed separately in Herceg (2009). Therefore, user testing of both content extraction and language ID is required in order to anticipate performance and assess the risk.[9] The user must consider (a) the specific use case (including file types), and (b) the capabilities and limitations of the content extractor and language ID products available to the given project.

6. Electronic Text and Machine Translation

Machine translation software automatically translates a document in a source language into a document in a target language. Depending on the use of the translated output, the translation could be useful as plain electronic text, or mirror the source document in a rich media file format. Typical use cases include the following:

- Use case: The user has one or more rich media files. Generate a plain text translation in the target language that is suitable for search engine indexing.
- Use case: The user has one or more rich media files. Generate a rich media file translation of each that preserves the formatting of the source.

The latter use case is more complicated than the former. In this case, the MT software has to perform file format parsing or content extraction in order for the MT to operate on the electronic text. And it has to track the offsets where the formatting (structural elements) of the electronic text occurs so that it can be reinserted into the translated result. This process is further complicated when rich formatting occurs within each phrase or sentence to be translated. Because languages have different properties, it can be challenging to determine where to segment the source electronic text and where to reinsert formatting in the target.

Each machine translation product has its preferred file types, so product inputs and outputs vary. Language Weaver and Systran are example commercial products.

[8] The extent of the language-specific operations applied to the electronic text depends on the richness of the features in the search engine.
[9] A discussion with a technical vendor contact can reveal the software's architecture and elicit the cases in which a content extractor and/or language ID would fail.

7. Electronic Text and Computer Assisted Translation

Computer Assisted Translation (CAT) software helps human translators perform translation work faster and more accurately by reusing previous translation work, managing terminology, and automating the target document formatting, among other features. Previous translation work is reused by retrieving full sentence matches from a repository called *translation memory* (TM). The following is a typical use case.

- Use case: The user has a document in a rich media file. Allow the user to translate the document sentence by sentence, focusing on the electronic text alone, and have the software automatically generate a rich media output file that reflects the formatting of the source.

The CAT software must perform a level of content extraction or file format parsing in order to obtain electronic text. Access to the source document's electronic text is required for sentence segmentation, and for matching full sentences against the TM.

There are at least two reasons that CAT software needs reliable electronic text from the source file. First, retrieval of relevant translations from the TM depends on a source sentence derived from reliable electronic text. When there are errors in the source text, the CAT software may not return good TM matches. Second, users commit sentence translations to the TM for future retrieval and reuse by other users. The source sentence for each committed entry must be reliable electronic text for future TM queries to succeed. Therefore, the user of CAT technology must consider the file formats supported by the CAT software and the quality of the electronic text that the CAT software will extract. Often, this can only be determined through a well-designed experiment or test.

8. Electronic Text and Text Analytics

Reliable electronic text is important to *text analytics*, which is an emerging label for a vast spectrum of tools focused on knowledge discovery in documents (Feldman, 2004; Grimes, 2007). Many of these applications are discussed in Feldman and Sanger (2007). We have chosen to discuss the following four related areas, influenced by these authors:

- Information extraction (including named entity recognition)
- Text categorization (topics, subjects; includes text filtering)[10]
- Text mining
- Visualization and browsing

Information extraction (IE) and text categorization focus on deriving structured data from unstructured electronic text. Such information can then be added to a database for a variety of uses, including data mining. Information extraction processes unstructured electronic text to identify, or tag, entities (e.g., person, location, organization), attributes (i.e., qualifying details/features of entities), facts (e.g., relationships), and events (Feldman & Sanger, 2007, p. 96). Text categorization involves tagging electronic text documents (or textual objects) according to a predefined taxonomy of terms (p. 64). Commercial information extraction tools and named entity recognizers include BBN Identifinder, Basis Technology Rosette Entity Extractor, and SAP Text Analysis/Annotation Manager. The performance of these human language technologies depends on the integrity of the electronic text on which they operate.

[10] *Text clustering*, also called *document clustering*, performs unsupervised grouping and can be used when the user chooses to not specify a priori categories.

Text mining is essentially data mining on the structured data derived from what Feldman and Sanger (2007) call *text mining preprocessing techniques*.[11] These techniques include information extraction and text categorization, and the many component NLP technologies typically used inside information extraction and text categorization applications, such as tokenization, part-of-speech tagging, syntactic parsing, and shallow parsing (pp. 57, 60-61). For example, a database of structured data derived from unstructured customer comments can be mined for patterns of positive or negative sentiment. Discovered patterns can be validated and a classifier can be trained to locate the same patterns in related, yet unseen electronic text data. Commercial text mining applications include SAS Text Miner and IBM SPSS Modeler. Again, quality electronic text is essential for text mining to perform well.

Visualization and browsing also builds on structured data from text mining preprocessing techniques. The structured data can be used to relate, or join, documents together. These relationships provide paths that can be traversed, on demand, given a user interface. There are a number of innovative ways to display, or visualize, for users, collections of documents based on features and inter-document relationships at varying levels of granularity (e.g., tables, graphs, charts, maps). With an innovative user interface and display in hand, a user can browse an entire document collection. Pacific Northwest National Laboratory's IN-SPIRE is an example of such a visualization tool. Reliable electronic text is key to visualization and browsing; without it, users will be navigating a mass of spurious data.

It is common to find these types of text analytics tools combined with a number of the aforementioned human language technologies. For example, structured data added to search applications allows faceted search, pattern matching, and name matching. Visualization and browsing can be instrumental in aiding the discovery of patterns for text mining. Again, reliable electronic text is a prerequisite.

When electronic text resides in rich media files, a content extractor and/or language and encoding ID software are prerequisites and must dependably generate reliable electronic text. The user of text analytics software must consider the expected quality of the electronic text contained in his/her files, and the file formats supported by the text analytics software. Electronic text side effects caused by poor content extraction, language ID, and encoding ID, will cause text analytics software to fail or generate erroneous results that are misleading and should not be added to a database. Therefore, it is important for the user of text analytics software to test and validate its performance for a given use.

Table 1 shows a wide variety of forms in which text analytics can appear.

[11] The topic of data mining is beyond the scope of this paper. For a detailed treatment of data mining, see Witten & Frank (2005).

Table 1. Use cases for processing a collection of text-containing files.

Use Case	Software Type
Visually highlight the person names, organizations, and locations in a collection of documents.	Information Extraction, Named Entity Recognition
Identify from the electronic text the relationships between named entities and any events described.	Information Extraction
Develop structured data from the electronic text using NER, IE and text categorization, visualize that data in a number of presentation types (tables, graphs, charts, maps), and allow visual browsing.	Visualization and browsing
Add various category metadata to each file based on its electronic text, so that the files can be prioritized by category (i.e., tagging for prioritization).	Text Categorization
Add spam category metadata to each email based on its electronic text, so that spam can be deleted.	Text Categorization: Spam Filtering
Add structured metadata to each file based on NER, IE, and text categorization processing of the electronic text. Index the structured data along with the electronic text. And allow users to browse by structured data and use that data to iteratively refine queries.[12]	Text Categorization: Faceted Search
Alert users by email (or update an RSS feed) when the electronic text of a new document matches a predefined, important query.	Text Categorization: Alerting
Process the electronic text with NER, IE, and text categorization to generate structured metadata. Apply pre-trained statistical models for discovering interesting patterns in new data. Identify new predictors for interesting patterns in the structured data and allow the user to train new statistical models for these patterns.	Text Mining
Process the electronic text with NER, IE, and text categorization to generate structured metadata. Apply pre-trained statistical models for discovering interesting patterns in new data. Visualize that data in a number of presentation types (tables, graphs, charts, maps), and allow visual browsing. Identify new predictors for interesting patterns in the structured data and allow the user to train new statistical models for these patterns.	Text Mining, Visualization, and Browsing

[12] Examples include Endeca and FAST Search Server 2010 for SharePoint.

We already mentioned a few vendors of text analytics software. Vendors with more resources can bundle together a handful of text analytics applications, whereas smaller vendors may only offer one of these text analytics types. Larger vendors may have the resources to bundle together and embed content extraction, language ID, and encoding ID, whereas smaller firms may not have this luxury, and may only be able to process the electronic text in a very limited number of file types. Prior to using text analytics software, it is incumbent upon the user to investigate the software's supported file types, extent of content extraction (including language and encoding ID), and performance on test data, and, to match findings to the specific use case at hand.

9. Summary

We discussed the importance of dependable text extraction, and reliable electronic text, to a variety of human language technologies. And we equipped the reader to be discerning about using human language technologies with rich media files. We presented typical use cases for each technology area—search, machine translation, computer assisted translation, and text analytics—and highlighted the various electronic text side effects to look for when using content extractors or file converters: mishandling or loss of structural elements, character data, and/or textual objects. In many cases, users and developers need dependable content extraction, language ID, and encoding ID, or must ensure that sufficient capabilities reside in the HLT software being used. Often, the only way to determine that a given human language technology will dependably process electronic text is through in-house experimentation and testing, using representative data sets. We refer the reader to Herceg (2007) for guidance on such technology evaluations.

References

Adobe (2008). Document management — Portable document format — Part 1: PDF 1.7. Retrieved June 16, 2010 from http://www.adobe.com/devnet/acrobat/pdfs/PDF32000_2008.pdf

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. New York: ACM Press.

Carrier, B. (2009). Extracting searchable text from Arabic PDFs. Basis Technology. Retrieved September 15, 2010 from http://www.basistech.com/knowledge-center/forensics/extracting-text-from-Arabic-PDF.pdf

Condon, S. L., Herceg, P. M., Phillips, J., & Rubenstein, A. H. (2005). Assessment of software to identify language and encoding of digital documents. Technical Report MTR080119. McLean, VA: The MITRE Corporation.

English, B., Alderman, B., & Ferraz, M. (in press). *Microsoft SharePoint 2010 Administrator's Companion*. Redmond, WA: Microsoft Press.

Feldman, R. (2004). ACM Thirteenth Conference on Information and Knowledge Management (CIKM) CIKM and Workshops 2004, Tutorials, Text Analytics: Theory and Practice. Retrieved September 9, 2010 from http://www.ir.iit.edu/cikm2004/tutorials.html#T2

Feldman, R. & Sanger, J. (2007). *The text mining handbook: Advanced approaches in analyzing unstructured data*. Cambridge: Cambridge University Press.

File Format. (n.d.). In *Wikipedia*. Retrieved September 14, 2010 from http://en.wikipedia.org/wiki/File_format.

Gospodnetić, O., & Hatcher, E. (2005). *Lucene in action*. Greenwich, CT: Manning Publications, Co.

Grimes, S. (2007, February 8). Defining text analytics. Message posted to http://intelligent-enterprise.informationweek.com/blog/archives/2007/02/defining_text_a.html

Herceg, P. M. (2007). Defining useful technology evaluations. Technical Report MTR070061R1. McLean, VA: The MITRE Corporation.

Herceg, P. M. (2009). The content extraction technology gap: Accessing the textual content of computer files. Technical Report MTR090437. McLean, VA: The MITRE Corporation.

Herceg, P. M., Allison, T. B., & Ball, C. N. (2009). A MITRE search prototype using Outside In and Lucene. Technical Report MTR090085. McLean, VA: The MITRE Corporation.

King, J. (2008, July 29). Text content in PDF files. Article posted to http://blogs.adobe.com/insidepdf/2008/07/text_content_in_pdf_files.html (Adobe Blogs).

Lawrence, G., Kehoe, W., Oya R., Walters, W., Kenney, A. (2000). Risk management of digital Information: A file format investigation. Retrieved August 6, 2010 from http://www.clir.org/pubs/reports/pub93/pub93.pdf (Council on Library and Information Resources).

Magic Number. (n.d.). In *Wikipedia*. Retrieved October 16, 2009 from http://en.wikipedia.org/wiki/Magic_number_(programming).

Rauch, C., & Rauber, A. (2004). Preserving digital media: Towards a preservation solution evaluation metric. In Proceedings of the 7th International Conference on Asian Digital Libraries (ICADL 2004) (pp. 203–212). Berlin, Heidelberg: Springer. Retrieved August 6, 2010 from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.87.6646&rep=rep1&type=pdf

Waters, D., & Garrett, J. (1996). *Preserving Digital Information, Report of the Task Force on Archiving of Digital Information, The Commission on Preservation and Access and The Research Libraries Group*. Retrieved September 2, 2010 from http://www.clir.org/pubs/reports/pub63watersgarrett.pdf (Council on Library and Information Resources).

Witten, I., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques, Second Edition*. San Francisco: Morgan Kaufmann.